CLAIMS

What is claimed is:

1. A server capable of receiving requests from at least one client, said server comprising:

a queue for handling requests received from said at least one client;

a queue manager that removes at least the next-to-be handled request from said queue when the number of requests in said queue exceeds a request threshold value.

2. The server of claim 1, wherein said server is connected to said client through a network.

3. The server of claim 2, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or asynchronous transfer mode (ATM) network.

4. The server of claim 1, wherein said queue is a first-in first-out queue.

5. The server of claim 1, wherein said queue is a virtual queue.

6. The server of claim 1, wherein said request threshold value is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

7. The server of claim 6, wherein said response time is an elapsed time from the receipt of said client request until the time said server completes handling of said client request.

8. The server of claim 1, wherein said request threshold value is a first predetermined number that is larger than the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

9. The server of claim 8, wherein said response time is an elapsed time from the receipt of said client request until the time said server completes handling of said client request.

10. The server of claim 1, wherein, prior to processing said queue, said queue manager:

determines if there are requests in said queue and, if that determination is not true, waits until a request is present;

determines if the number of requests in said queue exceeds said request threshold value, and, if that determination is true, repeatedly removes the request

in the first slot of said queue and advances the remaining requests in said queue by one position until the number of requests is less than or equal to said request threshold value; and

processes the request stored in the first slot of said queue.

11. The server of claim 10, wherein said queue manager terminates if no requests are found in said queue.

12. The server of claim 1, wherein, prior to processing said queue, said queue manager:

determines if there are requests in said queue and, if that determination is not true, waits until a request is present;

determines if the number of requests in said queue exceeds said request threshold value, and, if that determination is true, removes from said queue a number of requests equal to the difference between the actual number of requests in said queue and said request threshold value, and advances the remaining requests said queue by a number of positions equal to the difference between the actual number of requests in said queue and said request threshold value; and

processes the request stored in the first slot of said queue.

13. The server of claim 12, wherein said queue manager terminates if no requests are found in said queue.

14. The server of claim 8, wherein, prior to processing said queue, said queue manager:

determines if there are requests in said queue and, if that determination is not true, waits until a request is present;

determines if the number of requests in said queue exceeds said first predetermined number, and, if that determination is true, repeatedly removes the request in the first slot in said queue and advances the remaining requests in said queue by one position until the number of requests in said queue is less than a second predetermined number, which is smaller than said first predetermined number of requests; and

processes the request stored in the first slot of said queue.

15. The server of claim 14, said queue manager terminates if no requests are found in said queue.

16. The server of claim 14, wherein said second predetermined number is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

17. The server of claim 8, wherein, prior to processing said queue, said queue manager:

determines if there are requests in said queue and, if that determination is not true, waits until a request is present;

determines if the number of requests in said queue exceeds said first predetermined number, and, if that determination is true, removes from said queue a number of requests equal to the difference between the actual number of requests in said queue and a second predetermined number which is smaller than said first predetermined number, and advances the remaining requests said queue by a number of positions equal to the difference between the actual number of requests in said queue and said second predetermined number; and

processes the request stored in the first slot of said queue.

18. The server of claim 17, said queue manager terminates if no requests are found in said queue.

19. The server of claim 17, wherein said second predetermined number is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

15. A method for handling overload of requests in a client-server environment, said server having a request queue and a request manager, wherein:

said request manager determines if there are requests in said queue and, if that determination is not true, waits until a request is present;

if said request manager determines that the number of requests in said queue exceeds said request threshold value, said request manager repeatedly removes the request in the first slot of said queue and advances the remaining

requests in said queue by one position until the number of requests is less than or equal to said request threshold value; and

said request manager processes the request stored in the first slot of said queue.

21. The method of claim 20, wherein said request manager terminates if no requests are found in said queue.

22. The method of claim 20, wherein said request threshold value is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

23. The method of claim 20, wherein said requests are sent from said client to said server over a network.

24. The method of claim 23, wherein said network a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

25. The method of claim 20, wherein said queue is a first-in first-out queue.

26. The method of claim 20, wherein said queue is a virtual queue.

27. A method for handling overload of requests in a client-server environment, said server having a request queue and a request manager, wherein:

said request manager determines if there are requests in said queue and, if that determination is not true, waits until a request is present;

if said request manager determines that the number of requests in said queue exceeds said request threshold value, said request manager removes from said queue a number of requests equal to the difference between the actual number of requests in said queue and said request threshold value, and advances the remaining requests said queue by a number of positions equal to the difference between the actual number of requests in said queue and said request threshold value; and

said request manager processes the request stored in the first slot of said queue.

28. The method of claim 27, said request manager terminates if no requests are found in said queue.

29. The method of claim 27, wherein said request threshold value is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

30. The method of claim 27, wherein said requests are sent from said client to said server over a network.

31. The method of claim 30, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

32. The method of claim 27, wherein said queue is a first-in first-out queue.

33. The method of claim 27, wherein said queue is a virtual queue.

34. A method for handling overload of requests in a client-server environment, said server having a request queue and a request manager, wherein:

said request manager determines if there are requests in said queue and, if that determination is not true, waits until a request is present;

if said request manager determines that the number of requests in said queue exceeds said first predetermined number, said request manager repeatedly removes the request in the first slot of said queue and advances the remaining requests in said queue by one position until the number of requests in said queue is less than a second predetermined number, which is smaller than said first predetermined number of requests; and

said request manager processes the request stored in the first slot of said queue.

35. The method of claim 34, said request manager terminates if no requests are found in said queue.

36. The method of claim 34, wherein said second predetermined number is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

37. The method of claim 34, wherein said requests are sent from said client to said server over a network.

38. The method of claim 37, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

39. The method of claim 34, wherein said queue is a first-in first-out queue.

40. The method of claim 34, wherein said queue is a virtual queue.

41. A method for handling overload of requests in a client-server environment, said server having a request queue and a request manager, wherein:

said request manager determines if there are requests in said queue and, if that determination is not true, waits until a request is present;

said request manager determines if the number of requests in said queue exceeds said first predetermined number, and, if that determination is true, removes from said queue a number of requests equal to the difference between the actual number of requests in said queue and a second predetermined number which is smaller than said first predetermined number, and advances the remaining requests said queue by a number of positions equal to the difference between the actual number of requests in said queue and said second predetermined number; and

said request manager processes the request stored in the first slot of said queue .

42. The method of claim 41, said request manager terminates if no requests are found in said queue.

43. The method of claim 41, wherein said second predetermined number is the result of an average response time of said server to a client request divided by an average time used said server to process said client request.

44. The method of claim 41, wherein said requests are sent from said client to said server over a network.

45. The method of claim 44, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

46. The method of claim 41, wherein said queue is a first-in first-out queue.

47. The method of claim 41, wherein said queue is a virtual queue.

48. A computer software product for handling overload of requests in a client-server environment, said server having a request queue and a request manager, said computer software product comprises:

software instructions for enabling said server, said request queue and said request manager to perform predetermined operations, and a computer readable medium bearing the software instructions, said predetermined operations comprise:

determining if there are requests in said queue and, if that determination is not true, waiting until a request is present;

if said predetermined operations determine that the number of requests in said queue exceeds said request threshold value, said predetermined

operations repeatedly removing the request in the first slot of said queue and advancing the remaining requests in said queue by one position until the number of requests is less than or equal to said request threshold value; and

processing the request stored in the first slot of said queue.

49. The computer software product of claim 48, said request manager terminates if no requests are found in said queue.

50. The computer software product of claim 48, wherein said request threshold value is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

51. The computer software product of claim 48, wherein said requests are sent from said client to said server over a network.

52. The computer software product of claim 51, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

53. The computer software product of claim 48, wherein said queue is a first-in first-out queue.

54. The computer software product of claim 48, wherein said queue is a virtual queue.

55. A computer software product for handling overload of requests in a client-server environment, said server having a request queue and a request manager, said computer software product comprises:

software instructions for enabling said server, said request queue and said request manager to perform predetermined operations, and a computer readable medium bearing the software instructions, said predetermined operations comprise:

determining if there are requests in said queue and, based upon that determination, waiting until a request is present;

if said predetermined operations determine that the number of requests in said queue exceeds said request threshold value, said predetermined operations removing from said queue a number of requests equal to the difference between the actual number of requests in said queue and said request threshold value, and advancing the remaining requests said queue by a number of positions equal to the difference between the actual number of requests in said queue and said request threshold value; and

processing the request stored in the first slot of said queue.

56. The computer software product of claim 55, said request manager terminates if no requests are found in said queue.

57. The computer software product of claim 55, wherein said request threshold value is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

58. The computer software product of claim 55, wherein said requests are sent from said client to said server over a network.

59. The computer software product of claim 58, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

60. The computer software product of claim 55, wherein said queue is a first-in first-out queue.

61. The computer software product of claim 55, wherein said queue is a virtual queue.

62. A computer software product for handling overload of requests in a client-server environment, said server having a request queue and a request manager, said computer software product comprises:

software instructions for enabling said server, said requestqueue and said request manager to perform predetermined operations, and a computer readable medium bearing the software instructions, said predetermined operations comprise:

determining if there are requests in said queue and, based upon that determination, waiting until a request is present;

if said predetermined operations determine that the number of requests in said queue exceeds said first predetermined number, said predetermined operations repeatedly removing the request in the first slot of said queue and advancing the remaining requests in said queue by one position until the number of requests in said queue is less than a second predetermined number, which is smaller than said first predetermined number of requests; and

processing the request stored in the first slot of said queue.

63. The computer software product of claim 62, said request manager terminates if no requests are found in said queue.

64. The computer software product of claim 62, wherein said second predetermined number is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

65. The computer software product of claim 62, wherein said requests are sent from said client to said server over a network.

66. The computer software product of claim 65, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

67. The computer software product of claim 62, wherein said queue is a first-in first-out queue.

68. The computer software product of claim 62, wherein said queue is a virtual queue.

69. A computer software product for handling overload of requests in a client-server environment, said server having a request queue and a request manager, said computer software product comprises:

software instructions for enabling said server, said request queue and said request manager to perform predetermined operations, and a computer readable medium bearing the software instructions, said predetermined operations comprise:

determining if there are requests in said queue and, if that determination is not true, waits until a request is present;

determining if the number of requests in said queue exceeds said first predetermined number, and, if that determination is true, removing from said queue a number of requests equal to the difference between the actual number of requests in said queue and a second predetermined number which is smaller than said first predetermined number, and advancing the remaining requests said queue by a number of positions equal to the difference between the actual number of requests in said queue and said second predetermined number; and

processing the request stored in the first slot of said queue.

70. The computer software product of claim 69, said request manager terminates if no requests are found in said queue.

71. The computer software product of claim 69, wherein said second predetermined number is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

72. The computer software product of claim 69, wherein said requests are sent from said client to said server over a network.

73. The computer software product of claim 72, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

74. The computer software product of claim 69, wherein said queue is a first-in first-out queue.

75. The computer software product of claim 69, wherein said queue is a virtual queue.

76. A computer system adapted for handling overload of requests in a client-server environment, the computer system comprising:

at least one client and at least one server, wherein said at least one client and said at least one server are coupled to each other;

a memory comprising software instructions adapted to enable the computer system to:

determine if there are requests in said queue and, if that determination is not true, wait until a request is present;

determine if the number of requests in said queue exceeds said request threshold value, and, if that determination is true, repeatedly remove the request in the first slot of said queue and advance the remaining requests in said queue by one position until the number of requests is less than or equal to said request threshold value; and

process the request stored in the first slot of said queue.

77. The computer system of claim 76, said software instructions are further adapted to terminate if no requests are found in said queue.

78. The computer system of claim 76, wherein said request threshold value is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

79. The computer system of claim 76, wherein said requests are sent from said client to said server over a network.

80. The computer system of claim 79, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

81. The computer system of claim 76, wherein said queue is a first-in first-out queue.

82. The computer system of claim 76, wherein said queue is a virtual queue.

83. A computer system adapted for handling overload of requests in a client-server environment, the computer system comprising:

at least one client and at least one server, wherein said at least one client and said at least one server are coupled to each other;

a memory comprising software instructions adapted to enable the computer system to:

determine if there are requests in said queue and, based upon that determination, wait until a request is present;

determine if the number of requests in said queue exceeds said request threshold value, and, if that determination is true, remove from said queue a number of requests equal to the difference between the actual number of requests in said queue and said request threshold value, and advance the remaining requests said queue by a number of positions equal to the difference between the actual number of requests in said queue and said request threshold value; and

process the request stored in the first slot of said queue.

84. The computer system of claim 83, said software instructions are further adapted to terminate if no requests are found in said queue.

85. The computer system of claim 83, wherein said request threshold value is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

86. The computer system of claim 83, wherein said requests are sent from said client to said server over a network.

87. The computer system of claim 86, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

88. The computer system of claim 83, wherein said queue is a first-in first-out queue.

89. The computer system of claim 83, wherein said queue is a virtual queue.

90. A computer system adapted for handling overload of requests in a client-server environment, the computer system comprising:

at least one client and at least one server, wherein said at least one client and said at least one server are coupled to each other;

a memory comprising software instructions adapted to enable the computer system to:

determine if there are requests in said queue and, based upon that determination, wait until a request is present;

determine if the number of requests in said queue exceeds said first predetermined number, and, based on that determination, repeatedly remove

the request in the first slot of said queue and advance the remaining requests in said queue by one position until the number of requests in said queue is less than a second predetermined number, which is smaller than said first predetermined number of requests; and

process the request stored in the first slot of said queue.

91. The computer system of claim 90, said software instructions are further adapted to terminate if no requests are found in said queue.

92. The computer system of claim 90, wherein said second predetermined number is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

93. The computer system of claim 90, wherein said requests are sent from said client to said server over a network.

94. The computer system of claim 93, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

95. The computer system of claim 90, wherein said queue is a first-in first-out queue.

96. The computer system of claim 90, wherein said queue is a virtual queue.

97. A computer system adapted for handling overload of requests in a client-server environment, the computer system comprising:

at least one client and at least one server, wherein said at least one client and said at least one server are coupled to each other;

a memory comprising software instructions adapted to enable the computer system to:

determine if there are requests in said queue and, if that determination is not true, waits until a request is present;

determine if the number of requests in said queue exceeds said first predetermined number, and, if that determination is true, remove from said queue a number of requests equal to the difference between the actual number of requests in said queue and a second predetermined number which is smaller than said first predetermined number, and advance the remaining requests said queue by a number of positions equal to the difference between the actual number of requests in said queue and said second predetermined number; and

process the request stored in the first slot of said queue.

98. The computer system of claim 97, said software instructions are further adapted to terminate if no requests are found in said queue.

99. The computer system of claim 97, wherein said second predetermined number is the result of an average response time of said server to a client request divided by an average time used by said server to process said client request.

100. The computer system of claim 97, wherein said requests are sent from said client to said server over a network.

101. The computer system of claim 100, wherein said network is a local area network (LAN), a wide area network (WAN), an Infiniband network or an asynchronous transfer mode (ATM) network.

102. The computer system of claim 97, wherein said queue is a first-in first-out queue.

103. The computer system of claim 97, wherein said queue is a virtual queue.